Emotions and the Engineering of Adaptiveness in Complex Systems

M.G. Sánchez-Escribano*, R. Sanz

Autonomous System Laboratory, Universidad Politécnica de Madrid, C/ José Gutiérrez Abascal 2, 28006, Madrid, Spain.

Abstract

A major challenge in the engineering of complex and critical systems is the management of change, both in the system and in its operational environment. Due to the growing of complexity in systems, new approaches on autonomy must be able to detect critical changes and avoid their progress towards undesirable states. We are searching for methods to build systems that can tune the adaptability protocols. New mechanisms that use system-wellness requirements to reduce the influence of the outer domain and transfer the control of uncertainty to the inner one.
Under the view of cognitive systems, biological emotion suggests a strategy to configure value-based systems to use semantic self-representations of the state. A method inspired by emotion theories to causally connect the inner domain of the system and its objectives of wellness, focusing on dynamically adapting the system to avoid the progress of critical states. This method shall endow the system with a transversal mechanism to monitor its inner processes, detecting critical states and managing its adaptivity in order to maintain the wellness goals. The paper describes the current vision produced by this work-in-progress.

Keywords: Model-Driven Engineering, Runtime Model, Cognitive System, Emotional Models, Context-Awarenes

1. Introduction

Complex technical systems inevitably cause the emergence of complex engineering problems. Most of them are distributed systems-of-systems formed by different parts that cannot be functionally isolated but performing

* *E-mail address:* mguadalupe.sanchez@upm.es

federated work and sharing resources[1]. A major challenge of *systems engineering* is the building of dependable systems, regardless the operational environment or any unexpected events not predicted at design time. Complex systems can be regarded as *Open Systems* under the conception advanced by von Bertalanffy[2], i.e. "systems that maintain themselves in exchange of materials with the environment, and in continuous building up and breaking down of their components". This produces the emergence of *inner domains*, i.e. multiple states and service patterns that become a complex set of relevant circumstances that also influence the system. This concept of *domain* is a central aspect of the vision proposed in this paper. Furthermore, the external environment in which systems perform their missions is also harsh and full of unexpected events. Global uncertainty in unstructured environments and unpredictable internal and external events, cannot be fully considered in the design phases. At design time, the information about the operation of the system is partial, and the model for controlling behavior and acceptable changes is incomplete. However, the way in which complex systems deal with these unplanned events strongly influences its final resilience and robustness. This scenario reveals the need for designing novel methods to meet the requirements of complex systems, and autonomous self-organization is a key strategy for this end.

Ashby[3] explored several *Principles of the Self-Organizing System*, where "systems in general go to equilibrium" by the *Spontaneous Generation of Reorganization* among their multiple parts. Every transition from any state to one equilibrium state, requires the selection of proper states that determine the decisive stability. The challenge of building dependable complex systems should be addressed by designing systems with the capability to select those proper states, and this directly implies the design of systems with deep plasticity and self-awareness. The exploration of self-adaptive capabilities becomes a key matter and the *engineering of adaptiveness*, as the development of systematic methods to endow systems with adaptivity, becomes a critical need in complex systems engineering.

1.1. Models, adaptation and cognition background

How to encompass the influence of *change* wherever it comes from, is one of the main issues of software engineering for self-adaptive systems[4]. *Autonomic Computing* (AC)[5] is one of the new paradigms addressing those exigent needs from the viewpoint of self-adapting software. It deals with the self-management ability of distributed resources to attain system-level goals. *Self-adaptive software* aims to adjust different artifacts or attributes in response to changes in the *self* and in the *context* of a software system. The domain of *Self-X systems* addresses the architectural aspects of systems that are autonomous in the management of certain architectural traits[6,7]. The area of *Model-Integrated Computing* (MIC)[8] studies how to extend the scope and use of models to maintain a correlated sequence of changes between the model itself and the complex system in which it is embedded. A main feature of MIC is that it tries to match modeling paradigms to the needs of the systems engineering domain. *Model Driven Software Development* (MDSD)[9,10] focuses on using software models for increasing the quality and improving the effectiveness of the software process. A more open-domain line of research is what is referred as *Model Driven Enginering* (MDE)[11]. Within this area, *models@runtime* [12, 13] creates models used to reason about the system and its operating environment at run time[14]. Lastly, the dynamic identification of *changing requirements* is one of the new research challenges to assure the reliability of systems under unplanned or unexpected occurrences[4].

The approach suggested in this paper addresses the issue from the analysis of the cognitive architecture of biological emotion. There are plenty of examples of cognitive architectures that address partial aspects of human cognition[15,16]. From a more systems centric view, examples go from studies about *end-means patterns* as hierarchical organizations for structuring goals and means, to a large number of applicable analyses about complexity in engineering systems[3,17], organized systems[18], risk management[19] or entropy growth management, i.e. adjustment of structural artifacts to respond to the challenges of the environment[20]. The focus on emotion is not about the perceivable aspects of it[21] but about the structural traits concerning the adaptation of system structure to shape behavior[22]. All of this conforms a wide field of fuzzy subfields. We will focus on those paradigms of relevance for this paper such as self-adaptiveness, model centric and cognitive architectures.

1.2. Research framework and contribution

The issues discussed in this paper are part of wider research addressing the improvement of autonomy in technical systems. We call this effort the *ASys Project*. We are using a bio-inspired approach for the rigorous study of *Engineered Complex Adaptive Systems* (ECASs). We are referring to systems that can be classified at complexity level IV under the perspective of Magee and Weck[1], i.e. "plant, equipment and complex machines viewed as complicated systems that contain several parts constituting a functional and spatial unity". The *ASys Project* is a long-term research plan to design universal technologies for the construction of autonomous systems in any domain and with any level of autonomy. The approach is the identification of the core architectural patterns that can sustain the desired universality while being operationalisable in the construction of reusable assets for real world systems.

The present paper describes research-in-progress on the engineering of emotions. It is an interconnected study of engineered systems and biological emotion to break through the problem of autonomy and dynamical evolving of adaptiveness. This work suggests that emotional-based operativeness allows for a) a dynamical evaluation of references that the system uses for reasoning and for reconfiguring adaptiveness and b) some meaning of the inner context of the system to allow for self-representation. In this paper, we propose to build models that provide causal connections with the inner context of the system at runtime. We further propose to unfold the system into two longitudinal and transversal subsystems so as to manage the external and the inner context in a concurrent way. This scheme draws the distribution of control for the management of goals causality, allowing adaptiveness to evolve on the basis of its own references. The solution provides an internal space for decision-making and value-based reasoning (an internal space for adaptiveness), as well as two kinds of self-representation (regarding the state and its meaning). We suggest that autonomy requires not only self-reconfiguration capabilities to attain the expected outcomes at runtime, but also the capability to self-identify other non-defined outcomes aiming at reducing the drift to undesirable states. The paper describes the rationale for looking at emotion when searching for technological robustness (Section 2) the challenge of adaptiveness in terms of system requirements (Section 3), the role of model-based cognitive architectures (Section 4) and finally our proposed method to evolve adaptivity (Section 5). A conclusions section ends this work.

2. Cognition, Emotion and the Path to Technology

The more the complexity of modern technical systems grows, the more they get closer to the complexity of biological agents. Engineers look for solutions in biological systems trying to find patterns in their designs to solve problems in artificial systems. The goal of this paper is not to offer a formal study on biological emotion, but some core principles must be stated to understand the main content of our approach.

Emotion is a complex phenomenon, with subtle interactions among subjective and objective factors[23]. Besides its multidimensional character, there is not much debate about considering emotion as a feature of living systems for adapting to their environment throughout value-based semantics. Under the perspective of cognitive systems, biological emotion constitutes the strategy to configure value-based systems using semantic self-representation of the agent state[24,25]. The cognitive perspective of emotion suggests that emotion is the result of patterns of evaluation of relevant stimuli[22] which are carried out by means of some essential bodily processes.

Homeostasis describes mechanisms that hold constant a controlled variable, by means of sensing its deviation from a setpoint and feeding-back to amend the error. The concept of *allostasis* refers to the ability for maintaining stability through goal-directed change[26], describing mechanisms that override local feedback to meet anticipated demands. The idea of *stability-through-change* refers to physiological changes that anticipate requirements on the basis of local feedback, and *action readiness* captures the idea of change to be prepared for action. Finally, *arousal* as the regulation strategy of activation, and *appraisal* as the regulation strategy of interpreting the meaning of a stimulus[27], are two key concepts to help focus the implementation of our solution.

Emotion-based approaches to a technology for adaptiveness can offer architectural resources to dynamically accomplish a trade-off between multiple conflicting goals, linking requirements of mission and system wellness. Note that biological emotion is fundamentally related to system wellness and not to externally imposed objectives. It derives in utility functions and causal models that relate both the operational domain, and the abstractly built inner-context of the system, allowing the emergence of a new *domain of concern* where monitoring the inner system is

central; especially when there are conflicts among required goals or different operational directives. To address new requirements stemming from the interdependence of the operational domain and the inner state of the system, the system architecture can exploit an operative model of causal relationships based on an emotional analogy. The system should 1) develop semantics about inner and outer state, 2) be able to maintain knowledge about goals, 3) implement methods to manage value, 4) address change, either current or anticipated, 5) be able to compare system and value-based semantics and 6) feedback meaning from stimuli to system.

3. The problem of System Adaptiveness

A proper definition of *software adaptibility* is the versatility of changing the system to accommodate the occurrence of change in the operational environment while maintaining the fulfillment of general system goals specified at design time. Systems have to meet requirements that commonly are grouped into two dimensions, i. e. *functional requirements* (FRs) related with the mission, and *non-functional requirements* (NFRs) defining the overall or specific qualities of the whole system and commonly associated with *quality of service* (QoS) factors.

Our research assumes systems with different levels of abstraction and a SoS architecture where subsystems use services from other subsystems. It will be named the *longitudinal system* in this paper. Each subsystem hosts programs to provide its own services so it may be viewed as an individual complex system with defined FRs and NFRs. Writing and differentiating FRs and NFRs is an intricate and widely studied task[28-31]. In order to theoretically describe our approach, we draw upon the NFR definition made by Chung[31]. Considering an FR as a mathematical function $f: I \rightarrow O$, a NFR is just about anything that addresses characteristics of f, I, O or relationships between I and O. Each requirement defines a set of objectives of accomplishment, so the success of any requirement is analyzed by measuring the progress towards the objectives that the requirement defines. We will also that an NFR is a finite set of FRs and additional association rules defined within quiescent models.

We also label the difference between the task towards which the system has been designed -named *extrinsic goals*, and the system requirements to fulfill these tasks -named *intrinsic goals*. In this work we will assume the set of *intrinsic goals* as a set of FRs and NFRs that will be monitored by means of quantifiable performance measures.

3.1. System, Robustness and Adaptation

In a general sense, system *adaptivity* is the capacity of performing structural change in order to fulfill fixed or changing requirements and tuning to new operating circumstances during the lifetime of the system[32]. The concept of *robustness* refers to the ability of a system to respond adequately under unanticipated runtime conditions. Usually it is integrated within a more general concept such as *dependability* as a composite of NFRs that encompass several other NFRs: *reliability*, *availability*, *robustness*, *fault-tolerance*, *survivability*, *safety*, and *security*. Note the difference that some authors make between *correctness* and *robustness*. Robustness deals with those run time circumstances that have not been captured in the requirements specification of a system or rightly captured in the models at design time, while *correctness* of a system englobs the fulfillment of defined requirements. The *adaptability* should be designed in order to accomplish also correctness and robustness. Our emotion-based approach is centered in extending the correctness of a system to improve the fulfillment of these defined requirements even with unplanned occurrences, i.e. increasing robustness by means of adaptivity for wellness.

3.2. Dynamic Evolving of Adaptiveness on the basis of Emotional Behavior

The challenge of building a method to regulate the fulfillment of system requirements regardless the aftereffects of changes, requires a previous analysis about the sort of information that might be useful for the system in these circumstances of change. Under the viewpoint of emotional theory the term *system wellness* captures a sense of positive-valued meaning that feed-back the system. It underlies some inner processes into which the system monitors and evaluates itself regarding its own operational processes and its environment. We apply this idea to drive the system towards *wellness-oriented* predictable responses when unexpected events happen. Since we cannot have protocols to specifically attend any unknown event, we shall build the system to perceive a non-defined anomaly and be able to drive itself towards safer states. In some sense, the system should be able to recognize some

value-based meaning about the anomaly and feed-back itself with this meaning, enabling the decision-making processes concerning its current capabilities and the value received.

This approach implies the need to design a reference-system over which to realize anomalies as non-identifiable critical changes, and to define at design time safe states concerning the required property of wellness. The solution firstly demands the definition of wellness directives able to drive the critical changes towards safer states. To this end, it should be defined new orthogonal vectors of FR-NFRs for safer states. These wellness directives shall be measured to monitor the state, the change and the gradient of change.

4. Modeling Foundations for System Adaptiveness

Oreizy argues that *self-adaptive* software modifies its own behavior in response to changes in its operating environment[33]. However, it is essential that system be aware of these changes. *Autonomic performance* is intimately connected with the capability of evolving from one configuration to another. It implies the need of having capabilities for reconfiguring, monitoring and reasoning. This work is based on a *cognitive science* approach and has to be translated to software engineering domains. With minor differences, there is some sort of consensus about the foundations of biological *readiness* and its relevance in the scheme of emotional operation. However, how engineering systems might implement internal action readiness is not a clear but intricate problem of modeling. Systems need models to incorporate usable knowledge concerning the structural and functional baselines of readiness operative.

Models are abstractions of reality that contain just the essential aspects of this reality concerning the system[34]. The function of a model regards both the interpretation and the understanding of the system, as well as the drawing of conclusions in the form of other subsequent and usable models. In the domain of *open systems*[2], and considering the functional correlation requirement between the complex system and the software embedded in it, we shall necessarily assume that this software also lives in an open environment. The property of openness introduces the requirement of uncertainty management that might come from internal emergent faults, design or construction errors. Whatever the source of uncertainty may be, it should be dynamically attended. Thus it is necessary to extend the applicability of models to the runtime phase. This work suggests specific classes of models widely used within the paradigm of *models@runtime* such as *decision*, *behavior*, *architecture* and *variability models*, to monitor not just the punctual state of the system, but also behavioral changes due to subsequent influences affecting the state.

4.1. Building context models that improve context-awareness and request-awareness

Even if it may seem obvious, a system cannot realize anything it has not been built to perceive. Since uncertainties are not devised, the system cannot directly act concerning them. However, we might consider the fact that the system might be aware of an unknown event since it has no protocol either to classify or manage it. In other words, we can always create a new possibility of *unknown circumstance* and implement a general protocol to deal with it.

On the other side, the domain includes the portion of the environment that is relevant for the system; any change in the domain brings about significant events that affect the system operation. The emergence of new requirements is due to the enactment of a new subset of the entire domain that becomes relevant for the system at a given time. Since the uncertainty cannot be managed from the very moment when it occurs, the system must allow the progress of the aftereffects within the system towards a better point from where to monitor them. As designers, we can have better knowledge about the operation and the internal structure of the system than about the same issues concerning the external environment. Thus, we can define and model an abstract *inner-domain* using specific measurements concerning the accomplishment of FRs and NFRs, i.e. relevant information for the system that the system is able to use.

Once a new domain of operation is established, the system has to be endowed with the necessary causality to use the information that comes from it. The *inner-context model* will provide the necessary modeling support for the representation of the causal relationships between the system and the inner-domain, i.e. the conditions and circumstances that are relevant to the set of valuable events of the system. In this way, the system can monitor the progress towards the accomplishment of requirements, as well as the patterns of evolution of those requirements.

4.2. Architectural Foundations: The Longitudinal and Transversal systems

We propose to organize the whole system in terms of two subsystems to concurrently manage the external and the inner context. The external context is managed by the *longitudinal system* –an engineering system as explained in section 3– and the *transversal system*, which is used to apply feedback from the inner domain of the system over the system itself. In this way, it might be able to perform suitable decision-making concerning the value-based meaning of the inner-domain state (see Fig. 1).

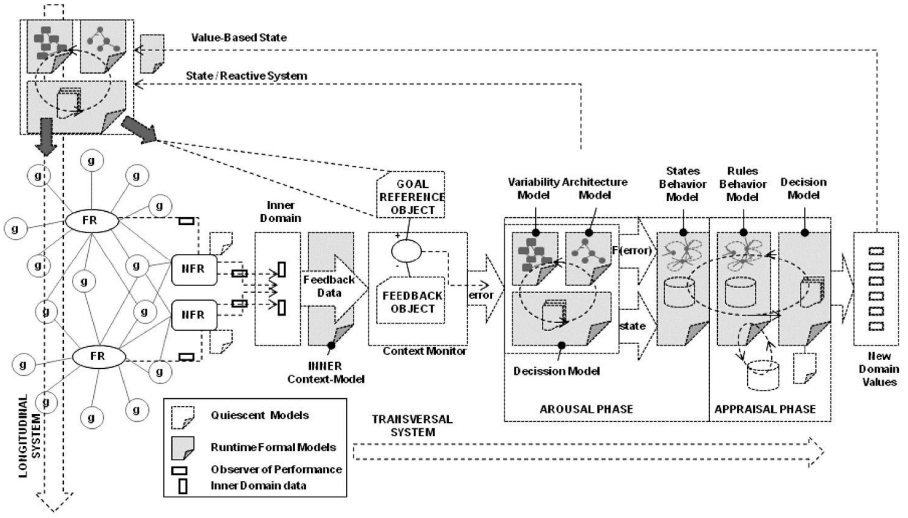

Fig. 1. Conceptual diagram of system operation

We envision the *longitudinal system* as a collection of distributed system-of-systems (SoSs) working federated to cope with general system objectives. The system as a whole should fulfill the *extrinsic goals* while maintaining the success of the *intrinsic goals*. The *transversal system* is an orthogonal system that monitors the *inner-context* of changes occurring inside the longitudinal system to produce usable action values in the form of system models. The transversal system will observe the domain of changes to afterwards transform them into usable models for the system, i.e. usable knowledge.

5. The Engineering of Evolving Adaptivity

Despite the fact that we draw analogies regarding the emotional operativeness, we assume that real processes of biological emotion do not exactly match with the methods here described. However, hereafter we will name the two stages of processing in an emotional analogous way: the *arousal* and *appraisal* phases of the emotion-based processing.

The longitudinal system measures the accomplishment of the intrinsic goals by means of some available metrics, i.e. *observers*, and sends the results of the measurements in the form of *feedback objects* to the transversal one. In the first phase, as in any other system, the transversal system receives data from its environment to then interpret it through the defined inner-context model. The transversal system also uses a setpoint that will serve to determine the accomplishment of objectives, i.e. the *goal-reference-object*. The difference between the values compared, will be an *error-object* that will affect the second phase of the transversal system: the emotion-based processing.

5.1. The Arousal-based strategy about Concerns

In analogy with the arousal process of emotion theories, this phase implements a regulation strategy to a) monitor and control the fulfillment of FRs and NFRs and b) activate patterns of rapid response concerning FR-NFR patterns that are critical for the system. The *error-object* affects a *variability model* built as a hierarchical set of features as proposed by Batory[35]. The feature model will dispose variability relationships (such as strongly required, weakly required, etc.) between units that represent FRs and NFRs. Changes in the variability model represent patterns of FRs and NFRs that constitute references used to estimate categories that directly might affect the longitudinal system. Examples of those categorizations are a) *pleasant/not pleasant* concerning the values of *error-object* with regard to the null value, b) *relevant/not relevant* concerning the value of error with regard to limit values, c) *severe/not severe* if these limits are surpassed, etc. This is done by means of a *decision model* and the *architectural model* of the longitudinal system regarding the patterns of affected components. Finally, the *decision model* also classifies the state of the system among those previously defined. To this end, the previous definition of states that relate the deviation of FR and NFR patterns and expected states for the system is required. What is relevant in this phase is the classification of *unknown-state* if it does not match any of those defined patterns.

5.2. The Appraisal-based strategy about Values

Analogously with the appraisal theories, this phase offers a regulation strategy by interpreting the meaning of these patterns of the arousal phase. To this end we envisage the need for two *behavioral models* as finite state machines to describe a) the behavior of plausible states and transitions relating to some parameters of relevance and b) the rules of interaction between the states of the system regarding some defined wellness.

The first model requires previous analysis at design time concerning the definition of parameters of relevance. These parameters will be built from the components of the vector of error (the *error-object*) with regard to values or combinations that might be critical, have continuous repetitions, etc. We envisage this model as a finite state machine defining the states of the system (including the *unknown-state*) and the transitions with regard the parameters of relevance. Afterwards, some previously defined emotional dimensions will form the domain of valuable states of the system. Once defined the emotional dimensions for the system (relevant orthogonal vectors of forward-direction or wellness directives), the interaction protocols between the states of the system will be modeled in order to analyze relevant gradients of change and relevant patterns stored in the past. This phase returns values concerning the *emotional state* of the system, using a *decision model* and the behavior of the two *behavior models*.

5.3. New Domain of Values

The returned *domain of values* will be modeled to be the core-domain model that represents the essence of the operation made by the transversal system. It will be used for major decision-making processes concerning the longitudinal system. Note that if emotions operate returning usable value for the system, this value will become a new abstract domain with additional semantics for the system. A new set of models in the longitudinal system will make the perception of the *domain of values* possible. Appropriate changes both over the longitudinal system as well as over the references used to compute the *error-object* in the transversal system will be applied.

4. Conclusions

We propose an emotion-inspired method to better manage uncertainty connecting local and wellness-based global decisions for complex systems at runtime. This paper has outlined a method to feedback the system with valuable meaning, allowing reasoning not only with regard to its adaptivity, but also with regard to dynamical evaluations concerning the inner state of the system to evolve the adaptivity. It draws a scheme to distribute the control in order to manage the causality of intrinsic goals, allowing dynamic adaptive capabilities on the basis of its own references for wellness. The method provides two kinds of self-representation regarding the state and its meaning, increasing at the same time the influence of the inner domain in decision-making processes. It is a more favorable scenario to control uncertainties since the aftereffects are transferred to the inner domain in the form of wellness requirements.

This work is an early step in our research, offering strategies regarding the improvement of environmental adaptiveness and hence improving system robustness. Even when it is not our current focus, an emotional system should be able to improve other aspects of the *dependability* of a system. We cannot discard solutions to improve *survivability* and some degrees of *reliability*. The task of the improvement of *safety* will likely be associated with an additional improvement of higher-order, model-based capabilities of reasoning, i.e. *cognitive* capabilities. For now, this work can be applied only at design time by modeling the relationships between the safety factors of some operational domain, and the operation of the system inside it. Nevertheless, we do not think that emotion-based technical systems should have improved *availability* beyond what the augmented robustness can offer. We are also aware of the many challenges that the implementation phase presents, including for instance those related to computational costs or to eventual inconsistencies that may come up while the system carries out the reconfigurations.

References

1. Magee, C.L., Weck, O.L.: Complex System Classification. Fourtheenth Annual International Symposium of the International Council On Systems Engineering (INCOSE); June 2004.
2. Bertalanffy, Ludwig von.: The Theory of Open Systems in Physics and Biology. Science. January 13, Vol. 111, 1950.
3. Ashby, W. Ross. Principles of the Self-Organizing System. In Heinz von Foerster and George W. Zopf (Eds.), Principles of Self-Organization Pergamon Press, 1962, pp. 255-78.
4. Cheng, B.H.C.; Lemos, R.; Giese, H.; Inverardi, P.; and Magee, J.; Software Engineering for Self-Adaptive Systems: A Research Road Map, Proceedings of Dagshtuhl Seminar on software engineering for self-adaptive systems, 2009.
5. Kephart, J.O.; Chess, D.M.: The Vision of Autonomic Computing. Computer, 36(1), 2003, pp. 41-50.
6. Sanz, R., Lopez, I., Bermejo-Alonso, J., Chinchilla, R., and Conde, R. Self-X: The control within. Proceedings of IFAC World Congress, 2005.
7. Babaoglu, O., Jelasity, M., Montresor, A., Fetzer, C., Leonardi, S., Moorsel, A., and Steen, M. Self-star Properties in Complex Information Systems: Conceptual and Practical Foundations. Springer-Verlag , 2005.
8. Sztipanovits, J. and Karsai, G.; Model-integrated computing, IEEE Computer, vol.30, pp. 100-111, 1997.
9. Greenfield, J.; Short, K.; Software factories: assembling applications with patterns, models, frameworks and tools, OOPSLA Companion, 2003, pp. 16-27.
10. Stahl, T.; Volter, M.; Model-driven Software Development, John Wiley, 2006.
11. Kent, S.; Model Driven Engineering, Proceedings of the Third International Conference On Integrated Formal Methods-IFM2002, Turku (Finland), May 15-17, pp. 286-298, 2002. LNCS 2335.
12. Fouquet, F., Nain, G., Morin, B., Daubert, E., Barais, O., Plouzeau, N. and Jèzéquel, J.M.; An Eclipse Modelling Framework Alternative to Meet the Models@Runtime Requirements. 15th International Conference, MODELS 2012, Innsbruck, Austria, September 30–October 5, 2012. Proceedings.
13. Daubert, E.; Fouquet, F.; Barais, O.; Nain, G.; Sunye, G.; Jezequel, J.-M.; Pazat, J-L; Morin, B., A models@runtime framework for designing and managing Service-Based Applications, 2012 Workshop on European Software Services and Systems Research, pp.10,11, 5-5 June 2012.
14. Blair, G.; Bencomo, N.; and France, R.B.; Models@run.time, Computer, 42(10), 2009, pp. 22-27.
15. Laird, J.E., A. Newell, and P. S. Rosenbloom. Soar: an architecture for general intelligence. Artificial Intelligence, 33(1):1–64, 1987.
16. Anderson, J. R. and C. Lebiere. The atomic components of thought. Lawrence Erlbaum Associates, Mahwah, N.J., 1998.
17. Axelrod and M. Cohen, HarnessingEngineering Principles and Practice, John Wiley & Complexity, Basic Books, New York, NY, 2000
18. March, J., Simon,H.; Organizations, 2nd Edition, Blackwell, Cambridge, MA, 1993.
19. Sage, A.P. ; Conflict and risk in systems management as complex adaptive systems issues. IEEE International Conference on Systems, Man, and Cybernetics, 2001 . 1767 - 1772 vol.3
20. Haghnevis, M., Askin, R.G.; A Modeling Framework for Engineered Complex Adaptive Systems. Systems Journal IEEE , 6 520-530 (2012)
21. Ekman, P. (1999). Basic emotions. In Dalgleish, T. and Power, M., editors, Handbook of Cognition and Emotion, chapter 3. Wiley, 1999.
22. Sanz, R., Sanchez-Escribano, M. G., and Herrera, C. A model of emotion as patterned metacontrol. Biologically Inspired Cognitive Architectures, Vol 1, No. 2, 2013.
23. Kleinginna, P.R., Jr. and Kleinginna, A.M.; A categorized list of emotion definitions, with a suggestion for a consensual definition. Motivation and Emotions, 5, 345-379 (1981)
24. Damasio, A.R. [1999] The feeling of what happens. New York: Harcourt-Brace & Company.
25. Picard, R. W.: What does it mean for a computer to "have" emotions?. Emotions in Humans and Artifacts, ed. By R. Trappl, P. Petta and S.
26. Sterling, P. and Eyer, J.; Allostasis: a new paradigm to explain arousal pathology. In S. Fisher and J. Reason (Eds.), Handbook of Life Stress, Cognition and Health (pp.629-649). New York: John Willey and Sons (1988)
27. Pessoa, L.; The Cognitive-Emotional Brain. The MIT Press 2013
28. Glinz, M.. On non-functional Requirements. 15th IEEE International Requirements Engineering Conference, 2007.
29. IEEE Recommended Practice for Software Requirements Specifications. IEEE Std. 830-1998, 1998.
30. Jacobson,I., G. Booch, and J. Rumbaugh. The Unified Software Development Process. Addison Wesley, 1999.